علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۷، شماره ۱، ۱۳۹۸
صفحات ۱۱۵- ۱۰۷
**مقاله عادی**

# موازی‌سازی کارا و تسریع عملیات انطباق رشته‌ها بر روی بستر پردازنده‌های چندهسته‌ای

میلاد غفوری[۱]، آرمین احمدزاده[۲]، سعید گرگین*[۳]

*نویسنده مسئول، دریافت: ۹۷/۰۶/۱۰، بازنگری: ۹۷/۱۲/۰۳، پذیرش: ۹۸/۰۴/۱۴

[۱،۲] پژوهشگر، علوم کامپیوتر، پژوهشکده علوم کامپیوتر، پژوهشگاه دانش‌های بنیادی، تهران، ایران
[۳] استادیار، مهندسی کامپیوتر، پژوهشکده برق و فناوری اطلاعات، سازمان پژوهش‌های علمی و صنعتی ایران، تهران، ایران

## چکیده

انطباق رشته‌ها یک بحث پایه‌ای و پرکاربرد در بیوانفورماتیک است که جهت یافتن میزان مشابهت توالی‌های آمینواسیدی و یا رشته-های دی.ان.ای از آن استفاده می‌شود. عمل انطباق دو رشته با یکدیگر یک عمل پایه‌ای است که در انطباق‌های چندگانه نیز از آن استفاده می‌شود. یکی از الگوریتم‌های انطباق دو رشته، الگوریتم نیدلمن-وانچ است که در آن از شیوه‌ی برنامه‌سازی پویا برای انجام این عملیات استفاده می‌شود. از چالش‌های مطرح در این الگوریتم، بالا بودن پیچیدگی زمانی آن است که جهت بهبود در سرعت اجرای این الگوریتم از راه‌کارهایی موازی می‌توان استفاده نمود. ازاین‌رو با ظهور پردازنده‌های چندهسته‌ای و امکان انجام محاسبات به شکل موازی می‌توان تسریع قابل‌ملاحظه‌ای را به دست آورد. بر اساس روش‌های موازی‌سازی موجود برای این الگوریتم، در هر گام خانه‌های آرایه‌ی به‌کاررفته در روش برنامه‌سازی پویا یک قطر به‌طور هم‌زمان و موازی تکمیل می‌شود. در این مقاله با یک تغییر دیدگاه نسبت به مسئله و در نظر گرفتن یک تصور گراف‌گونه، روشی ارائه شده است تا به نسبت روش‌های پیشین تسریع مناسبی را ارائه کند. نتایج به‌دست‌آمده نشان می‌دهد، در این پیاده‌سازی بهبود عملکردی تا حدود ۵,۹ برابر نسبت به پیاده‌سازی‌های پیشین حاصل می‌گردد..

**کلمات کلیدی:** الگوریتم نیدلمن-وانچ (Needleman-Wunsch)، انطباق رشته‌ها، بیوانفورماتیک، موازی‌سازی

## ۱- مقدمه

در انطباق دو رشته، سعی بر یافتن حروف مشابه بین رشته‌ها است و هرچه حروف مشابه بیشتری را بتوان با یکدیگر متناظر کرد یعنی انطباق بهتری از دو رشته به‌دست‌آمده است. البته، همان‌طور که در ادامه نیز نشان داده خواهد شد، تناظر بین حروف متفاوت امتیازهای متفاوتی دارد و درنهایت هدف بیشینه نمودن مجموع این امتیازها است. در انطباق چندگانه به یافتن رشته‌هایی با مشابهت بالا در بین چندین رشته پرداخته می‌شود. انطباق جفت رشته‌ها که پایه‌ی عملیات انطباق چندگانه را تشکیل می‌دهد [۱]، در علوم زیستی کاربردهای فراوانی دارد. به‌طور مثال، برای یافتن میزان مشابهت توالی آمینواسیدهای دو پروتئین به کار می‌رود که از انطباق آن‌ها برای تشخیص ساختار و سیر تکاملی جایگاه‌های آمینواسیدها در مجموعه‌ای از توالی‌ها استفاده می‌شود. لازم به ذکر است که پروتئین‌ها، پلیمر آمینواسید هستند و یک توالی از آمینواسیدها در کنار یکدیگر یک پروتئین را تشکیل می‌دهد؛ لذا در رشته‌های پروتئینی، آمینواسیدها نقش حروف این رشته‌ها را ایفا می‌کنند. از نظر تنوع، در طبیعت بیست نوع آمینواسید موجود است که وارد سنتز پروتئین می‌شوند[۲،۳،۴]. هر آمینواسید با یک نماد مشخص و برای نوشتن توالی پروتئین این نمادها پشت سر هم نوشته می‌شوند. همچنین انطباق توالی‌های پروتئینی برای کارهایی نظیر «مدل‌سازی همسان»[۱] و «پیش‌بینی بخش فعال»[۲] به کار می‌رود [۱]. ابزارهای مربوط به این عملیات نیاز به دقت بالایی دارند زیرا خطای ناشی از انطباق غیر دقیق باعث می‌شود مدل‌های پیش‌بینی بیوانفورماتیکی کیفیت قابل قبولی نداشته باشند. در مدل‌سازی همسان با داشتن یک پروتئین ناشناس، با استفاده از عملیات انطباقِ رشته‌ها می‌توان تشخیص داد که این پروتئین به کدام خانواده پروتئینی مشابهت بیشتری دارد و به عملکرد آن پی برد. اما در پیش‌بینی بخش فعال فقط بخشی از یک پروتئین برای مقایسه انتخاب می‌شود و کارکرد آن را به دست می‌آورند[۵]. همچنین انطباق یک جفت رشته برای به دست آوردن

میزان مشابهت دو رشته دی. ان. ای. کاربرد دارد. به قسمت‌های خاصی از کل دی. ان. ای. انسان که حاوی اطلاعات مهمی هستند ژن گویند. با استفاده از عملیات انطباقِ این بخش‌های مهم با دی. ان. ای. مرجع، می‌توان اختلالات افراد را تشخیص داد. همچنین از عمل انطباق رشته‌های دی. ان. ای. می‌توان برای شناسایی افراد استفاده کرد. برای دیگر موجودات نیز کاربردهایی دارد از قبیل اینکه می‌توان تشخیص داد که دی. ان. ای. مختص چه موجودی است و یا موجود مربوطه چه قابلیت‌هایی دارد.

قبل از آنکه عملیات انطباق صورت گیرد، نیاز است که توالی یابی دی. ان. ای. انجام شود. از طرفی پایگاه داده‌های ژنتیکی که حاوی اطلاعات دی. ان. ای. های توالی یابی شده هستند، با استفاده از تکنولوژی‌های کنونی نظیر توالی یابی نسل جدید[۳] با نرخ نمایی در حال رشد هستند و همین امر باعث شده الگوریتم‌های بیوانفورماتیکی (مانند الگوریتم‌های انطباق) به گلوگاهی برای آنالیز حجم عظیم داده‌های تولیدی تبدیل شوند. باوجود استفاده از سخت‌افزارهای قدرتمند، همچنان سرعت پردازش داده‌ها نسبت به سرعت تولید داده‌ها بسیار کندتر است[۶]. لذا نیاز به ارائه الگوریتم‌های سریع‌تر و تسریع الگوریتم‌های موجود همواره به‌عنوان یک نیاز جدی احساس می‌شود.

همگام با پیشرفت تکنولوژی و ورود پردازنده‌های چندهسته‌ای که قادر به پردازش هم‌زمان و موازی چندین دستورالعمل هستند، جهت بهبود سرعت الگوریتم‌ها می‌توان از موازی‌سازی استفاده کرد. در ابزارهای محک آزمون مختلف ازجمله Rodinia، مجموعه‌ای از مسائل پرکاربرد به همراه کدهای پیاده‌سازی موازی آن‌ها، وجود دارد[۷،۸]. مسئله‌ی انطباق رشته‌ها نیز یکی از این مسائل پرکاربرد است که رویکردهای موازی‌سازی برای آن موردتوجه قرار گرفته و در ابزارهای محک آزمون آمده است. در این مقاله، با به‌کارگیری توان محاسباتی پردازنده‌های چندهسته‌ای، عملکرد الگوریتم نسبت به پیاده‌سازی موجود در ابزار محک آزمون Rodinia به‌طور قابل‌ملاحظه‌ای افزایش یافته است.

یکی از مشکلات روش‌های ارائه شده این است که در گام‌های اولیه و انتهایی موازی‌سازی به‌طور کامل از منابع استفاده نمی‌شود. به بیان دقیق‌تر در گام‌های ابتدایی بعضی از هسته‌های پردازشی پردازنده بی‌استفاده خواهند ماند. درحالی‌که در روش پیشنهادی با عنوان محاسبات دوطرفه، این مراحل به‌صورت هم‌زمان اجرا می‌شوند و درواقع تعداد این مراحل کاهش می‌یابد، بنابراین از هسته‌های پردازشی پردازنده استفاده‌ی بهتری خواهد شد. همچنین در راستای استفاده بهینه از حافظه و افزایش کارایی، از روش بلوک‌بندی در کار پیشنهادی استفاده شده است.

در ادامه، در بخش دوم مقدمات لازم، برای درک بهتر مطالب بخش‌های آتی، ارائه می‌شود. در بخش سوم، کارهای انجام شده در حوزه انطباق دو رشته، از دو دیدگاه الگوریتمی و پیاده‌سازی مورد مطالعه و بررسی قرار گرفته‌اند. در بخش چهارم رویکرد موازی پیشنهادی جهت تسریع الگوریتم نیدلمن- وانچ ارائه شده است. بخش پنجم به ارزیابی روش پیشنهادی و مقایسه آن با روش‌های پیشین پرداخته و درنهایت نتیجه‌گیری مقاله در بخش ششم ارائه شده است.

## ۲- پیش‌زمینه

در الگوریتم‌های ارائه شده برای بررسی میزان مشابهت دو رشته، به دو شیوه‌ی محلی[۴] و سراسری[۵] عمل می‌شود[۹]. شیوه‌ی سراسری بیشتر در مواردی که طول رشته‌های داده شده یکسان باشند به کار می‌رود و سعی می‌کند تمام حروف را در عملیات انطباق سهیم بداند. اما در روش‌های محلی، زیررشته‌هایی که امتیاز حاصل از انطباق آن‌ها بیشترین امتیاز ممکن را داشته باشد در عملیات انطباق سهیم هستند[۱۰].

نحوه‌ی محاسبه‌ی امتیاز معمولاً به این صورت است که به ازای انطباق هر دو حرف یکسان امتیازی مثبت، به ازای انطباق دو حرف متفاوت امتیازی منفی و به ازای رد کردن هر حرف امتیاز منفی ثابتی به‌عنوان جریمه در نظر گرفته می‌شود. (در مورد طریقه محاسبه امتیاز در بخش ۱-۱-۲ به‌صورت مفصل‌تر توضیح داده شده است)

به عبارتی انطباق محلی، همان انطباق سراسری است با این تفاوت که هنگام انطباق دو رشته، تنها زیررشته‌ای از دو رشته در نظر گرفته می‌شود. الگوریتم نیدلمن-وانچ (Needleman-Wunsch) [۱۳] یکی از الگوریتم‌هایی است که به‌صورت برنامه‌نویسی پویا برای انطباق سراسری به کار می‌رود. به دلیل شباهت ذکر شده بین شیوه‌ی محلی و سراسری و از طرف دیگر به دلیل اینکه الگوریتم نیدلمن-وانچ رویکردی مشابه الگوریتم‌های محلی نظیر Smith-Waterman دارد (هر دو به روش برنامه‌سازی پویا عمل می‌کنند) در الگوریتم‌های محلی نیز می‌توان از الگوریتم نیدلمن-وانچ استفاده نمود[۱۱].

**مثال ۱**: برای درک بهتر تفاوت این دو شیوه، دو رشته‌ی HEAGAWGHEE، PAWHEAE و جدول امتیاز BLOSUM62 که در شکل ۳ نشان داده شده است را برای امتیاز انطباق حروف درنظر بگیرید. درصورتی‌که به ازای رد کردن هر حرف، جریمه‌ی «۶-» امتیازی اعمال شود، در این صورت اگر حروف $\{A, W, H, E, E\}$ مطابق شکل ۱- (الف) در هر دو رشته (HEAGAWGHEE و PAWHEAE) با یکدیگر انطباق یابند یکی از بهترین انطباق‌های سراسری صورت گرفته است. که در این صورت پنج حرف متناظر با یکدیگر یافت شده است. اما در انطباق محلی دو رشته، به ترتیب زیررشته‌های AWGHE و AWHE از دو رشته وارد عملیات می‌شوند که در این صورت نیز حروف $\{A, W, H, E\}$ از این دو زیررشته مطابق شکل۱- (ب) با یکدیگر انطباق می‌یابند. بنابراین در این حالت چهار حرف متناظر با یکدیگر یافت شده است.

در ادامه‌ی این بخش، در ابتدا به‌منظور سهولت در تبیین و درک روش برنامه‌نویسی پویا، روشی بازگشتی معرفی می‌شود. پس‌ازآن روش برنامه‌سازی پویا که درواقع همان الگوریتم نیدلمن-وانچ است ذکر می‌گردد. سپس نحوه‌ی امتیازدهی و جدول امتیاز به‌کاررفته معرفی می‌شود. درنهایت نحوه‌ی یافتن مسیری که منجر به حصول امتیاز بیشینه می‌شود، شرح داده خواهد شد.

```
String_1:   A W G H E
            | |   | |
String_2:   A W - H E
```

(الف) انطباق سراسری

```
String_1:   H E A G A W G H E - E
                |     |   | |   |
String_2:   - P A - - W - H E A E
```

(ب) انطباق محلی

شکل ۱- نحوه‌ی انطباق دو رشته‌ی HEAGAWGHEE و PAWHEAE در دو شیوه‌ی سراسری و محلی.

### ۲-۱- روش بازگشتی

با استفاده از روشی بازگشتی می‌توان بهترین حالت انطباق دو رشته را به دست آورد[۱۲]. تابع بازگشتی به این صورت است که از انتهای دو رشته شروع کرده و دو کاراکتر انتهایی آن‌ها را در نظر می‌گیرد، در این صورت سه حالت ممکن است رخ دهد، یا دو کاراکتر را باهم تطبیق می‌دهد و یا کاراکتر مربوط به رشته‌ی اول را رد می‌کند (به‌عبارت‌دیگر کاراکتر رد شده با Null -که در اینجا به آن اصطلاحاً «گپ» گفته می‌شود- تطبیق یافته است.) و کاراکتر ماقبل کاراکتر رد شده در رشته‌ی اول را برای گام بعدی تحت نظر قرار می‌دهد و یا کاراکتر مربوطه به رشته‌ی دوم را رد

می‌کند و کاراکتر ماقبل کاراکتر رد شده در رشته‌ی دوم را برای گام بعدی تحت نظر قرار می‌دهد.

بنابراین با توجه به اینکه نیاز به محاسبه ماکزیمم امتیاز است، می‌توان رابطه‌ی بازگشتی را با محاسبه‌ی ماکزیمم امتیاز حاصل شده از سه حالت مذکور به‌صورت رابطه بازگشتی (۱) نوشت.

در رابطه‌ی (۱)، $X_i$ زیررشته‌ای از رشته‌ی $X$ است که شامل $i$ حرف ابتدایی آن رشته است و $Y_j$ زیررشته‌ای از رشته‌ی $Y$ است که شامل $j$ حرف ابتدایی آن رشته است.

$$NW(X_i, Y_j) = max\{ \\ NW(X_{i-1}, Y_{j-1}) + Score(x_i, y_j) \\ NW(X_i, Y_{j-1}) + gapPenalty \\ NW(X_{i-1}, Y_j) + gapPenalty \\ \} \qquad (۱)$$

همچنین $Score(x_i, y_j)$ نشان‌دهنده‌ی امتیاز حاصل از انطباق دو حرف انتهایی $x_i$ و $y_j$ از دو رشته‌ی $X_i$ و $Y_j$ بوده و $gapPenalty$ بیان‌گر خطای حاصل از رد کردن یکی از این دو حرف است. همچنین شبه کد این روش در الگوریتم-۱ آورده شده است که تابع در ابتدا باید کل دو رشته را به‌عنوان ورودی بگیرد بنابراین اگر طول دو رشته‌ی $X$ و $Y$ را به ترتیب $k$ و $l$ باشند، ورودی‌های ابتدایی تابع $X_k$ و $Y_l$ خواهند بود یعنی تابع را باید به‌صورت $NW(X_k, Y_l)$ فراخوانی کرد. همان‌طور که در الگوریتم ۱ مشخص شده تابع بازگشتی مادامی‌که طول هر دو رشته بزرگ‌تر از صفر باشد ادامه خواهد یافت و اگر طول یکی از رشته‌ها صفر شود باید کل حروف

الگوریتم ۱ – انطباق توالی‌ها به روش بازگشتی

```
Inputs: Two sequences X and Y
Outputs: Optimal alignment score of X and Y
1.  p ← gapPenalty
2.  NW (Xi, Yj):
3.    m ← length(X)
4.    n ← length(Y)
5.    If (m = 0 OR n = 0)
6.      Return p × max{m , n}
7.    End If
8.    If xm-1 = yn-1
9.      Return Score(xi, yj) + NW(Xi-1, Yj-1)
10.   ElseIf
11.     Return p + max{NW(Xi, Yj-1) , NW(Xi-1, Yj)}
12.   End If
```

رشته‌ی دیگر را رد کرد و به تعداد حروف رد شده جریمه‌ی رد کردن به امتیاز اعمال خواهد شد.

## ۲-۲- روش نیدلمن – وانچ

نیدلمن - وانچ روشی است که از طریق برنامه‌نویسی پویا با استفاده از یک آرایه دو بعدی، تمامی حالت‌های ممکن برای انطباق حروف دو رشته را دربرمی‌گیرد تا انطباقی با بیشینه امتیاز از دو رشته حاصل شود[۱۳]. درواقع در این روش، تمامی مقادیر تابع بازگشتی معرفی شده در بخش قبلی به ازای تمامی ورودی‌های ممکن برای تابع بازگشتی $NW()$، در یک آرایه ذخیره می‌شود.

شبه کد مربوط به این الگوریتم در الگوریتم ۲ آورده شده است. همان‌طور که در این شبه کد مشاهده می‌کنید پس از مقداردهی مقادیر سطر اول و ستون اول آرایه، مقادیر مابقی خانه‌های آرایه‌ی D با استفاده از الگوی مشابهی محاسبه می‌شوند. به این صورت که همانند وابستگی‌های موجود در ورودی‌های تابع بازگشتی که در بخش قبلی معرفی شد، هر خانه‌ی $D[i][j]$ از آرایه به سه خانه‌ی $\{D[i-1][j-1], D[i][j-1], D[i-1][j]\}$ و امتیازِ انتقال از سه خانه به این خانه وابسته است. مقدار موجود در هر خانه بیان‌گر بیشینه امتیاز کسب شده از آغاز تا آن بخش از انطباق دو رشته است. درنهایت مقدار به‌دست‌آمده برای آخرین خانه از آرایه (خانه‌ی $D[length(X)-1][length(Y)-1]$) بیان‌گر بیشترین امتیاز ممکنِ حاصل از انطباق دو رشته خواهد بود که نحوه‌ی به دست آمدن آن به شیوه‌ی بازگشت به عقب است که در ادامه این روش توضیح داده شده است.

الگوریتم ۲ – انطباق توالی‌ها به روش نیدلمن-وانچ

```
Inputs: Two sequences X and Y
Outputs: Optimal alignment score of X and Y
1.  p ← gapPenalty
2.  For (i From 0 To length(X)) Do
3.    D[i][0] ← p×i
4.  End Do
5.  For (j From 0 To length(Y)) Do
6.    D[0][j] ← p×j
7.  End Do
8.  For (i From 1 To length(X)) Do
9.    For (j From 1 To length(Y)) Do
10.     Match ← D[i-1][j-1] + Score(xi, yj)
11.     Delete ← D[i-1][j] + p
12.     Insert ← D[i][j-1] + p
13.     D[i][j] ← max{Match, Insert, Delete}
14.   End Do
15. End Do
16. Return D[length(X)-1][length(Y)-1]
```

**مثال ۲**: در انطباق سراسری دو رشته‌ی AWGHE و PAWHE به شیوه‌ی نیدلمن-وانچ اگر مقدار جریمه به ازای رد کردن هر حرف برابر ۶- امتیاز باشد و برای محاسبه‌ی امتیاز حاصل از انطباق هر دو حرف از جدول استفاده شود، برای محاسبه‌ی مقدار هر خانه از جدول باید به سه خانه‌ای که وابسته است توجه داشت. به‌عنوان‌مثال در اولین گام که خانه‌ی مربوط به دو حرف ابتدایی (حروف P و A) بررسی می‌شوند ($D[1][1]$) و در شکل ۲ (الف) با مستطیل زردرنگ مشخص شده است، سه حالت ممکنی که باید از آن‌ها مقدار بیشینه را برگزید عبارت‌اند از:

- حالت اول) دو حرف با یکدیگر منطبق شوند و امتیاز ۱- حاصل از انطباق با مقدار ۰ حاصل از خانه‌ی مبدأ ($D[0][0]$) جمع گردد.
- حالت دوم) حرف P با یک گپ منطبق شود و جریمه‌ای ۶- امتیازی با مقدار ۶- موجود در خانه‌ی مبدأ دوم ($D[1][0]$) جمع گردد.
- حالت سوم) حرف A با یک گپ منطبق شود و جریمه‌ای ۶- امتیازی با مقدار ۶- حاصل از خانه‌ی مبدأ سوم این خانه ($D[0][1]$) جمع گردد.

با توجه به سه مقدار به‌دست‌آمده $\{-12, -12, -1\}$ باید مقدار ۱- در این خانه به‌عنوان مقدار بیشینه حاصل از سه حالت ممکن انتخاب گردد. همین روال برای تمام خانه‌های خالی جدول طی می‌شود تا درنهایت مطابق شکل ۲(ب) برای خان0647ی آخر مقدار ۱۶ به دست آید که این مقدار بیان‌گر امتیاز حاصل از انطباق سراسری دو رشته است. درنهایت برای یافتن مسیر طی شده تا به دست آمدنِ امتیاز پایانی، می‌توان از روش بازگشت به عقب استفاده نمود. این مسیر با رنگ سبز در شکل ۲(ب) نشان داده شده است.

### ۲-۲-۱- طریقه امتیازدهی

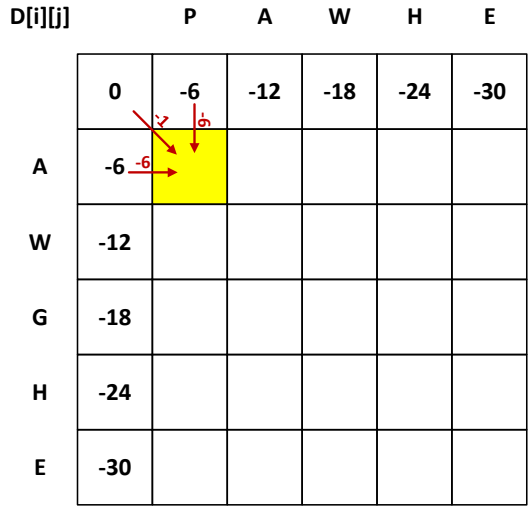
یکی از روش‌های بررسی میزان مطابقت دو رشته استفاده از جداول امتیاز است. در این جدول‌ها امتیاز انطباق به ازای هر دو آمینواسید قرار گرفته است. دسته‌ی خاصی از این جداول به BLOSUM معروف‌اند که نمونه‌ای از این جدول در شکل ۳ نشان داده شده است. جداول BLOSUM همراه با یک عدد بیان می‌شوند که این عدد بیان‌گر یک ویژگی آماری از امتیازهای موجود در آن جداول است. تجربه نشان داده BLOSUM62 بهترین جدول موجود برای تشخیص شباهت بین توالی‌های آمینواسیدی پروتئین‌ها است[۱۴]. هر یک از مقادیر موجود در این جدول بر اساس میزان تکرار وقوع جفت آمینواسید در پایگاه‌داده‌ی BLOCKS به دست می‌آید[۱۴].

همان‌طور که در شکل ۳ مشاهده می‌شود، با انطباق هر دو حرف یکسان امتیاز مثبتی حاصل می‌شود اما این امتیازها برای حروف متفاوت تغییر می‌کند، به‌عنوان‌مثال با انطباق دو حرف K با یکدیگر، ۵ امتیاز و با انطباق دو حرف H با یکدیگر، ۸ امتیاز حاصل می‌شود. همچنین در بعضی موارد با انطباق دو حرف متفاوت مانند K و R امتیاز مثبتی حاصل می‌گردد که بیان‌گر ناهمگن بودن این جدول است.

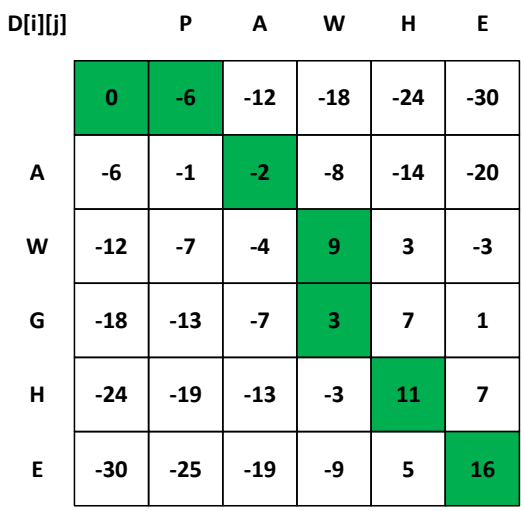
### ۲-۲-۲- گام بازگشت به عقب

همان‌طور که توضیح داده شده است، برای پیدا کردن مسیر از روش بازگشت به عقب استفاده می‌گردد. لذا با استفاده از روش بازگشت به عقب در الگوریتم نیدلمن-وانچ پس از یافتن بیشینه امتیاز به‌دست‌آمده، مسیر پیموده شده به دست می‌آید. مسیر پیموده شده بیان‌گر این است که کدام جفت از حروف دو رشته با یکدیگر انطباق یافته‌اند و کدام حروف با یک گپ انطباق یافته‌اند تا امتیاز بیشینه به دست آید.

برای یافتن مسیر پیموده شده، ابتدا از آخرین خانه‌ای که بررسی شده شروع کرده و خانه‌های موجود در مسیر رسیدن تا به نقطه‌ی آغازین رصد می‌شوند. شیوه‌ی این کار به این صورت است که برای یافتن خانه‌ی قبلی هر خانه مانند $\langle i,j \rangle$، هر سه خانه‌ای که امکان دسترسی به این خانه را داشته‌اند (خانه‌های $\{\langle i-1,j-1 \rangle , \langle i-1,j \rangle , \langle i,j-1 \rangle\}$) مورد بررسی قرار می‌گیرند. با در نظر گرفتن امتیاز موجود در هر یک از این سه خانه و همچنین امتیاز حاصل از انتقال هر یک از آن‌ها به خانه‌ی $\langle i,j \rangle$، خانه‌ای که مجموع دو عدد متناظرش با امتیاز موجود در $\langle i,j \rangle$ برابر باشد خانه‌ی قبلی $\langle i,j \rangle$ بوده است. بنابراین با استفاده از این روش می‌توان مسیر رسیدن از نقطه‌ی انتهایی به نقطه‌ی آغازین را رصد نمود.




(الف)




(ب)

شکل ۲ - نمایشی از آرایه‌ی به‌کاررفته در روش نیدلمن-وانچ. (الف) قبل از شروع عملیات. (ب) بعد از انجام عملیات (مسیر رصد شده تا نقطه‌ی شروع با رنگ سبز مشخص شده است.)

| | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V | B | Z | X | * |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 4 | -1 | -2 | -2 | 0 | -1 | -1 | 0 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 1 | 0 | -3 | -2 | 0 | -2 | -1 | 0 | -4 |
| R | -1 | 5 | 0 | -2 | -3 | 1 | 0 | -2 | 0 | -3 | -2 | 2 | -1 | -3 | -2 | -1 | -1 | -3 | -2 | -3 | -1 | 0 | -1 | -4 |
| N | -2 | 0 | 6 | 1 | -3 | 0 | 0 | 0 | 1 | -3 | -3 | 0 | -2 | -3 | -2 | 1 | 0 | -4 | -2 | -3 | 3 | 0 | -1 | -4 |
| D | -2 | -2 | 1 | 6 | -3 | 0 | 2 | -1 | -1 | -3 | -4 | -1 | -3 | -3 | -1 | 0 | -1 | -4 | -3 | -3 | 4 | 1 | -1 | -4 |
| C | 0 | -3 | -3 | -3 | 9 | -3 | -4 | -3 | -3 | -1 | -1 | -3 | -1 | -2 | -3 | -1 | -1 | -2 | -2 | -1 | -3 | -3 | -2 | -4 |
| Q | -1 | 1 | 0 | 0 | -3 | 5 | 2 | -2 | 0 | -3 | -2 | 1 | 0 | -3 | -1 | 0 | -1 | -2 | -1 | -2 | 0 | 3 | -1 | -4 |
| E | -1 | 0 | 0 | 2 | -4 | 2 | 5 | -2 | 0 | -3 | -3 | 1 | -2 | -3 | -1 | 0 | -1 | -3 | -2 | -2 | 1 | 4 | -1 | -4 |
| G | 0 | -2 | 0 | -1 | -3 | -2 | -2 | 6 | -2 | -4 | -4 | -2 | -3 | -3 | -2 | 0 | -2 | -2 | -3 | -3 | -1 | -2 | -1 | -4 |
| H | -2 | 0 | 1 | -1 | -3 | 0 | 0 | -2 | 8 | -3 | -3 | -1 | -2 | -1 | -2 | -1 | -2 | -2 | 2 | -3 | 0 | 0 | -1 | -4 |
| I | -1 | -3 | -3 | -3 | -1 | -3 | -3 | -4 | -3 | 4 | 2 | -3 | 1 | 0 | -3 | -2 | -1 | -3 | -1 | 3 | -3 | -3 | -1 | -4 |
| L | -1 | -2 | -3 | -4 | -1 | -2 | -3 | -4 | -3 | 2 | 4 | -2 | 2 | 0 | -3 | -2 | -1 | -2 | -1 | 1 | -4 | -3 | -1 | -4 |
| K | -1 | 2 | 0 | -1 | -3 | 1 | 1 | -2 | -1 | -3 | -2 | 5 | -1 | -3 | -1 | 0 | -1 | -3 | -2 | -2 | 0 | 1 | -1 | -4 |
| M | -1 | -1 | -2 | -3 | -1 | 0 | -2 | -3 | -2 | 1 | 2 | -1 | 5 | 0 | -2 | -1 | -1 | -1 | -1 | 1 | -3 | -1 | -1 | -4 |
| F | -2 | -3 | -3 | -3 | -2 | -3 | -3 | -3 | -1 | 0 | 0 | -3 | 0 | 6 | -4 | -2 | -2 | 1 | 3 | -1 | -3 | -3 | -1 | -4 |
| P | -1 | -2 | -2 | -1 | -3 | -1 | -1 | -2 | -2 | -3 | -3 | -1 | -2 | -4 | 7 | -1 | -1 | -4 | -3 | -2 | -2 | -1 | -2 | -4 |
| S | 1 | -1 | 1 | 0 | -1 | 0 | 0 | 0 | -1 | -2 | -2 | 0 | -1 | -2 | -1 | 4 | 1 | -3 | -2 | -2 | 0 | 0 | 0 | -4 |
| T | 0 | -1 | 0 | -1 | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 1 | 5 | -2 | -2 | 0 | -1 | -1 | 0 | -4 |
| W | -3 | -3 | -4 | -4 | -2 | -2 | -3 | -2 | -2 | -3 | -2 | -3 | -1 | 1 | -4 | -3 | -2 | 11 | 2 | -3 | -4 | -3 | -2 | -4 |
| Y | -2 | -2 | -2 | -3 | -2 | -1 | -2 | -3 | 2 | -1 | -1 | -2 | -1 | 3 | -3 | -2 | -2 | 2 | 7 | -1 | -3 | -2 | -1 | -4 |
| V | 0 | -3 | -3 | -3 | -1 | -2 | -2 | -3 | -3 | 3 | 1 | -2 | 1 | -1 | -2 | -2 | 0 | -3 | -1 | 4 | -3 | -2 | -1 | -4 |
| B | -2 | -1 | 3 | 4 | -3 | 0 | 1 | -1 | 0 | -3 | -4 | 0 | -3 | -3 | -2 | 0 | -1 | -4 | -3 | -3 | 4 | 1 | -1 | -4 |
| Z | -1 | 0 | 0 | 1 | -3 | 3 | 4 | -2 | 0 | -3 | -3 | 1 | -1 | -3 | -1 | 0 | -1 | -3 | -2 | -2 | 1 | 4 | -1 | -4 |
| X | 0 | -1 | -1 | -1 | -2 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -2 | 0 | 0 | -2 | -1 | -1 | -1 | -1 | -1 | -4 |
| * | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | -4 | 1 |

شکل ۳ - جدول BLOSUM62: امتیاز حاصل از انطباق هر دو آمینواسید

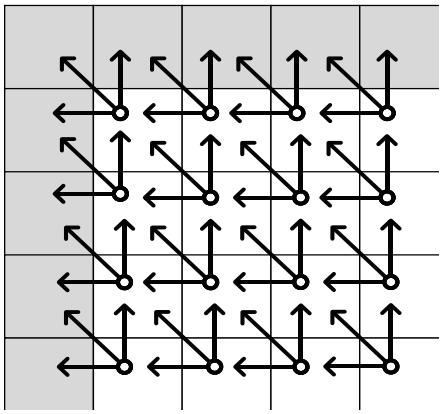
## ۳- کارهای انجام شده

با توجه به اهمیت این الگوریتم کارهایی برای تسریع آن انجام شده است، یکی از روش‌ها برای حل این مسئله به‌منظور کاهش پیچیدگی مکانی آن در حد خطی با استفاده از روش تقسیم و غلبه بوده است[۱۵]. یکی دیگر از رویکردها، استفاده از روابط جبرخطی برای نادیده گرفتن وابستگی بین داده‌ها در نقاط انفصال بخش‌های مختلف ماتریس است. به این صورت که تا جایی که امکان همگرا شدن حاصل‌ضرب بین ستون‌ها وجود دارد تعداد عملوندهای (ستون‌های) متوالی را می‌افزایند تا جایی که مقدار حاصل‌ضرب برابر یک شود. به عبارتی بین ستون ابتدایی و ستون انتهاییِ حاصل‌ضرب، وابستگی ضعیفی ایجاد شود. بنابراین نتایج به‌دست‌آمده از ستون انتهایی به بعد، نتایج صحیحی خواهند بود. نتایج ماقبل ستون آخر به این دلیل ناصحیح هستند که با یک ستون تصادفی شروع به پردازش ستون‌های مربوط به آن بخش کرده‌اند. این پیاده‌سازی‌ها که از ویژگی‌های جبری داده‌ها استفاده می‌کنند ممکن است در بدترین حالت خود در حد پیاده‌سازی سریال تنزل پیدا کنند. چراکه ممکن است در هنگام پیاده‌سازی موازی، پس از بخش‌بندی ماتریس به تعداد هسته‌های استفاده شده به‌منظور صحت داده‌های بخش قبلی ماتریس، عملیات مربوط به این بخش را به‌صورت مداوم تکرار کند. ازاین‌رو زمان اجرا برابر با زمان اجرای سریال این الگوریتم خواهد بود [۱۶،۱۷]. اما الگوریتم مناسب دیگری که برای موازی‌سازی ارائه شده به این صورت است که عملیات پر کردن خانه‌های موجود روی هر قطر را به‌صورت موازی انجام می‌دهد[۱۸] که اکثر مقاله‌ها از آن بهره برده اند، به‌عنوان‌مثال، پیاده‌سازی‌ای بر روی پردازنده‌های چندهسته‌ای به‌صورت خط لوله انجام شده که از دستور schedule (static,1) در OPENMP استفاده می‌کند[۱۹].

در روشی دیگر به‌منظور بهبود عملیات انطباق چندین جفت رشته، همین روش موازی‌سازی قطری را برای انطباق بین هر جفت رشته در نظر می‌گیرد و مضاف بر آن با دسته‌بندی رشته‌ها، عملیات انطباق هر جفت رشته را نیز در یک بلوک جداگانه از پردازنده‌های گرافیکی[۶] پردازش می‌کند[۲۰]. همان‌طور که در بخش ۲-۲ ذکر شد، وابستگی داده‌ای بین خانه‌های آرایه وجود دارد. در شکل ۴ وابستگی داده‌ای برای به دست آوردن بیشینه امتیاز عملیات انطباق بین دو رشته، نشان داده شده است. یعنی مقدار هر خانه به سه خانه‌ی بالا، چپ و بالا-چپ خود وابسته است. بنابراین در مرحله‌ی اول تنها خانه‌ی $\langle 0,0 \rangle$ است که به هیچ خانه‌ی دیگری وابسته نیست و مقدار مربوط به این خانه را می‌توان به دست آورد و در این خانه قرار داد. در گام بعدی، دو خانه‌ی مجاور این خانه یعنی خانه‌های $\{\langle 1,0 \rangle, \langle 0,1 \rangle\}$ هستند که به هیچ‌یک از خانه‌های خالی وابسته نیستند بنابراین مقادیر این خانه‌ها را نیز می‌توان به دست آورد و داخل این خانه‌ها قرار داد.

یکی دیگر از روش‌های ارائه شده، این است که آرایه را به بلوک‌هایی مربعی شکل با ابعاد یکسان تقسیم‌بندی کرد که هر یک از این بلوک‌ها تعداد یکسانی از خانه‌های آرایه را شامل شوند و سپس عمل موازی‌سازی روی این بلوک‌ها انجام شود (استراتژی بلوک‌بندی) و خانه‌های هر بلوک به‌صورت ترتیبی و توسط یک رشته‌نخ تکمیل گردند[۲۱]. همچنین در کار مقاله [۲۲]، الگوریتم مقایسه توالی‌ها با معماری انتقال پیام بر روی رایانه‌های موازی با معماری هرمی[۷] و مجازی پیاده‌سازی شده است که نشان داده شده که بهترین کارایی را نسبت به متدهای پیشین دارد. هرچند که فراهم آوردن بستر آن هزینه‌ی هنگفتی را در بر دارد، اما الگوریتم ارائه شده در این مقاله قابلیت ترکیب با پیاده‌سازی‌های پیشین را نیز خواهد داشت.

به همین ترتیب، در هر مرحله تنها خانه‌های یک قطر فرعی هستند که بدون وابستگی به خانه‌های فاقد مقدار، قابل مقداردهی هستند و می‌توان به‌صورت هم‌زمان آن‌ها را مقداردهی کرد. همان‌طور که در شکل ۵ نیز نمایش داده شده، در هر مرحله مقادیر وابسته خانه‌های یک قطر فرعی هستند که در گام‌های پیشین مشخص‌شده‌اند.



شکل ۴ - نمایش وابستگی‌های موجود در آرایه‌ی به‌کاررفته در روش نیدلمن وانچ

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | 3 | 4 | 5 | 6 | 7 |
| 3 | 4 | 5 | 6 | 7 | 8 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 5 | 6 | 7 | 8 | 9 | 10 |
| 6 | 7 | 8 | 9 | 10 | 11 |

شکل ۵ – نمایشی از ترتیب انجام عملیات موازی‌سازی

## ۴- روش پیشنهادی

### ۴-۱- موازی‌سازی دوطرفه

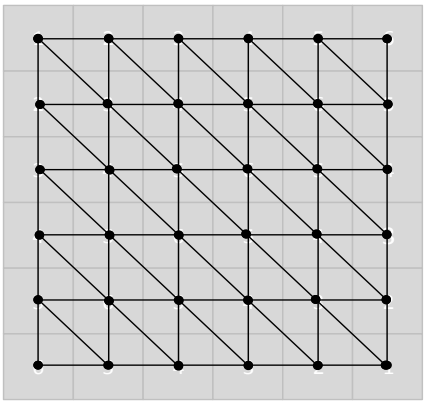
در بخش قبلی ذکر شد که برای موازی‌سازی الگوریتم نیدلمن- وانچ، در هر گام خانه‌های موجود بر روی یک قطر فرعی، به‌صورت هم‌زمان پر می‌شوند. یکی از مشکلات این روش این است که در گام‌های اولیه و انتهایی موازی‌سازی به‌طور کامل از منابع استفاده نمی‌شود. به بیان دقیق‌تر در گام‌هایی که تعداد خانه‌های روی یک قطر کمتر از تعداد پردازنده‌ها باشند، بعضی از هسته‌های پردازشی پردازنده بی‌استفاده خواهند ماند. درحالی‌که در روش پیشنهادی در این بخش، بعضی از این مراحل هم‌زمان باهم اجرا می‌شوند و درواقع تعداد این مراحل کاهش می‌یابد و همچنین، تعداد خانه‌های هر گام از مراحل موازی‌سازی که با یکدیگر به‌طور موازی پُر می‌شوند دو برابر شده، بنابراین از هسته‌های پردازشی پردازنده استفاده‌ی بهتری خواهد شد. ازاین‌رو با توجه به افزایش تعداد هسته‌های فیزیکی موجود در پردازنده‌ها و استفاده غیر بهینه از آن‌ها در روش پیشین دلیلی بر ارائه روش پیشنهادی است که در همان ابتدای اجرای الگوریتم، تعداد هسته‌های فعال بیشتری به اجرای الگوریتم تخصیص داده شود.

در الگوریتم-۳ شبه کد مربوط به این روش آورده شده است. مطابق این الگوریتم موازی‌سازی از دو سر ابتدایی و انتهایی جدول انجام می‌شد که در خطوط ۷ تا ۱۰ خانه‌های ابتدایی جدول به‌صورت پیشرو و در خطوط ۱۱ تا ۱۴ خانه‌های انتهایی جدول به‌صورت پس‌رو پر می‌شوند. برای شرح مفصل‌تر و درک بهتر این روش باید مشابه شکل ۶، یک تصور گرافی از این جدول داشت به‌طوری‌که خانه‌های جدول به‌عنوان گره‌ها و امتیازهای انتقال بین دو خانه‌ی مجاور به‌عنوان وزن یال‌های آن

باشند. درنتیجه با این فرض، هدف یافتن مسیری با بیشینه وزن ممکن از نقطه‌ی $\langle 0,0 \rangle$ به نقطه‌ی $\langle n,m \rangle$ از گراف خواهد بود. بنابراین می‌توان هم‌زمان با حرکت از نقطه‌ی $\langle 0,0 \rangle$ به پایین، از نقطه‌ی $\langle n,m \rangle$ نیز با بالا حرکت کرد، به‌عنوان‌مثال در گام اول هم‌زمان با نقطه‌ی $\langle 0,0 \rangle$ نقطه‌ی $\langle n,m \rangle$ از آرایه نیز تکمیل می‌شود.

الگوریتم ۳ – موازی‌سازی دوطرفه‌ی نیدلمن–وانچ

```
Inputs: Two sequences X and Y
Outputs: Optimal alignment score of X and Y
1.  p ← gapPenalty
2.  Nd ← numberOfDiagonalSteps
3.  For (d From 1 To Nd) Do
4.    Parallel For (cell in CellsWithIdx(d)) Do
5.      i ← cell.row
6.      j ← cell.col
7.      If tracingDirection(cell) = FORWARD
8.        Match ← D[i-1][j-1] + Score(xi, yj)
9.        Delete ← D[i-1][j] + p
10.       Insert ← D[i][j-1] + p
11.     Else
12.       Match ← D[i+1][j+1] + Score(xi, yj)
13.       Delete ← D[i+1][j] + p
14.       Insert ← D[i][j+1] + p
15.     End If
16.     D[i][j] ← D[i][j] + max{Match, Insert, Delete}
17.   End Do
18. End Do
19. Return max{CellsWithIdx(Nd)}
```



شکل ۶ - تبدیل فضای آرایه‌ی به‌کاررفته به فضای گرافی گونه

در گام بعدی هم‌زمان با مقداردهی نقاط $\langle 0,1 \rangle$ و $\langle 1,0 \rangle$ در قطر دوم از بالا نقاط $\langle n\ ,\ m-1 \rangle$ و $\langle n-1\ ,\ m \rangle$ که در قطر دوم از پایین قرار دارند نیز مقداردهی می‌شوند و به همین ترتیب ادامه داده تا درنهایت این دو پیمایش تداخل یابند. در این گام، مجموع امتیازهای به‌دست‌آمده از دو مسیر بالایی و پایینی برای هر یک از نقاط قطر فرعی را به دست آورده و درنهایت بیشینه مقدار به‌دست‌آمده در بین نقاط این قطر به‌عنوان بیشینه امتیاز کسب شده از تطابق دو رشته انتخاب می‌شود. این گام از اجرای الگوریتم پیشنهادی در راستای تعیین نتیجه به اجرای الگوریتم اضافه شده است که نسبت به استفاده بهینه از هسته‌های پردازشی در اجرای الگوریتم، پیچیدگی زمانی چندانی ایجاد نکرده است. چراکه روند اجرای آن به‌صورت موازی است و از طرفی دیگر پیچیدگی که این گام در اجرای الگوریتم پیشنهادی اضافه می‌کند برابر با اندازه قطر اصلی جدول است که در روش‌های پیشین نیز این بار محاسباتی وجود داشته است.

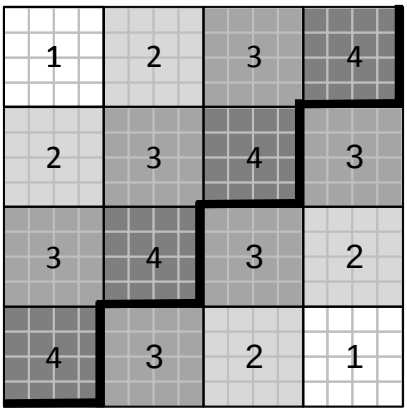
در شکل ۷ شمای مناسبی از روش موازی‌سازی پیشنهادی آورده شده است. در این شکل خانه‌هایی که به‌صورت هم‌زمان پر می‌شوند با یک رنگ و یک عدد یکسان مشخص شده‌اند. به تعبیری دیگر، علاوه بر بررسی زیررشته مشترک از آغاز دو رشته (بررسی پیشرو)، به‌صورت موازی از انتهای آن‌ها نیز شروع به بررسی می‌شود (بررسی پس‌رو). در این صورت باید حالت‌های بسیاری بررسی شوند که این دو پویش در کدام نقطه از دو رشته یکدیگر را ملاقات خواهند کرد که اجرای این روند زمان‌بر خواهد شد. ازآنجاکه تعداد حالت‌های ممکن زیاد است، این حالت‌ها به حالت‌هایی محدود می‌شوند که در آن‌ها تعداد عملیات هر دو پویش به یک تعداد باشند تا بیشترین میزان موازی‌سازی انجام شود. برای این منظور، (با فرض اینکه طول هر دو رشته برابر $n$ باشد) حالت‌ها این‌گونه خواهند بود که در تمامی آن‌ها هر یک از پویش‌ها (پیشرو و پس‌رو) $n$ حرف از مجموع تعداد حروف بررسی شده از هر دو رشته را پویش کرده باشند. به عبارتی اگر در بررسی پیشرو $n_0$ حرف از رشته‌ی اول و $n_1$ حرف از رشته‌ی دوم پویش شده باشند و مابقی حروف رشته‌ها توسط بررسی پس‌رو پویش گردند، باید رابطه‌ی $n_0+n_1=n$ برقرار باشد. در این صورت در جدولی که برای محاسبه‌ی ماکزیمم امتیاز استفاده می‌شود، محل تلاقی دو پویش روی بزرگ‌ترین قطر فرعی در نظر گرفته می‌شود. بدین ترتیب نیمه‌ی بالایی جدول توسط بررسی پیشرو و نیمه‌ی پایینی جدول توسط بررسی پس‌رو تکمیل خواهند شد و درنتیجه دو پویش تعداد عملیات برابری را انجام خواهند داد.

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | 3 | 4 | 5 | 6 | 5 |
| 3 | 4 | 5 | 6 | 5 | 4 |
| 4 | 5 | 6 | 5 | 4 | 3 |
| 5 | 6 | 5 | 4 | 3 | 2 |
| 6 | 5 | 4 | 3 | 2 | 1 |

شکل ۷– نحوه‌ی موازی‌سازی در روش پیشنهادی

## ۴-۲-استفاده از استراتژی بلوک‌بندی

همان‌طور که در بخش قبل بیان شد، یکی از روش‌های تسریع استفاده از استراتژی بلوک‌بندی است. در استراتژی بلوک‌بندی آرایه به بلوک‌هایی مربعی شکل با ابعاد یکسان تقسیم‌بندی می‌شود، به‌طوری‌که هر یک از این بلوک‌ها تعداد یکسانی از خانه‌های آرایه را شامل می‌شوند. با اعمال این استراتژی بر روی این کار، در هر گام از موازی‌سازی خانه‌های داخل بلوک‌های یک قطر هم‌زمان باهم پر می‌شوند و خانه‌های موجود در هر بلوک به‌صورت ترتیبی و توسط یک رشته نخ تکمیل می‌گردند. برای مثال در گام اول، یک رشته نخ، خانه‌های موجود در بلوک گوشه‌ی بالا-چپ جدول و یک رشته نخ خانه‌های موجود در بلوک گوشه‌ی پایین-راست جدول را محاسبه و پر می‌کنند. در گام دوم، خانه‌های بلوک‌هایی که روی دو قطری که در شکل ۸ با عدد ۲ مشخص شده‌اند و از اطلاعات به‌دست‌آمده در گام اول استفاده می‌کنند، پر می‌شوند. توجه شود که در این گام چهار رشته‌نخ به کار گرفته می‌شوند چراکه به ازای هر بلوکی که در شکل ۸ با عدد ۲ نمایش داده شده است، یک رشته‌نخ استفاده می‌شود و همین روال تا جایی که به بزرگ‌ترین قطر فرعی که آخرین قطر باقیمانده است برسد ادامه می‌یابد. در این گام خانه‌های بلوک‌های این قطر توسط پویش پیشرو پر می‌شوند. درنهایت نیاز به به دست آوردن حاصل‌جمع پویش پیشرو و پس‌رو مربوط به هر یک از مسیرها است. در شکل ۸، محل تقاطع دو مسیر پیشرو و پس‌رو روی نوار مشکی مشخص شده قرار دارد. بنابراین برای به دست آمدن مجموع امتیازهای هر مسیر، کافی است حاصل‌جمع هر یک از اعداد خانه‌هایی که روی نوار مشکی قرار دارند با خانه‌های زیرینشان به دست آید. بیشترین حاصل‌جمع مابین این اعداد، همان مسیر با بیشترین امتیاز در گراف است و برای به دست آوردن خود مسیر باید از نقطه‌ی متناظر با آن نقطه شروع به ردیابی مسیر نماییم.




شکل ۸– نحوه‌ی موازی‌سازی در روش پیشنهادی هنگام استفاده از استراتژی بلوک‌بندی

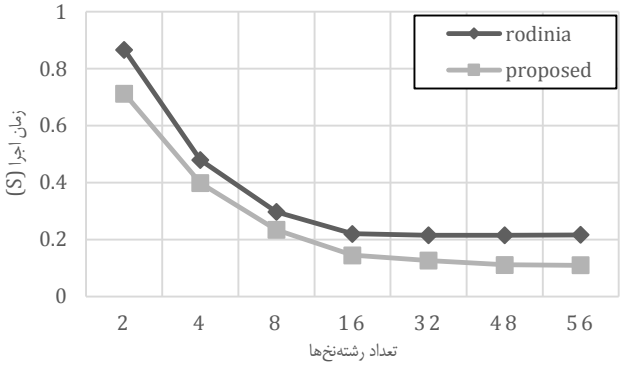
## ۵- مقایسه و ارزیابی

نتایجی که در ادامه آمده نتایج آزمایش‌هایی است که با استفاده از کامپایلر gcc v5.3.1 بر روی دو پردازنده‌ی Intel® Xeon® Processor E5-2697 v3 به‌دست‌آمده است. این دو پردازنده روی‌هم‌رفته ۲۸ هسته دارند. همچنین در آزمایش‌های انجام شده از جدول امتیاز BLOSUM62 به‌عنوان جدول امتیاز تطابق بین حروف استفاده شده است. در ابزار محک آزمون Rodinia، مجموعه‌ای از مسائل پرکاربرد به همراه کدهایی که به‌صورت موازی برای آن‌ها پیاده‌سازی شده است، وجود دارد. موضوع تطابق دو رشته یکی از این موارد است. نتایجی که در ادامه خواهید دید حاصل مقایسه‌ای است بین زمان اجرای روش ارائه شده و روش پیاده‌سازی شده در نسخه‌ی ۳,۱ از ابزار محک آزمون Rodinia [۲۳] برای بخشی از کد که به یافتن بیشینه امتیاز به‌دست‌آمده می‌پردازد. همچنین داده‌های به‌کاررفته در تمام آزمایش‌ها بر اساس تکه کدی از پیاده‌سازی Rodinia که به‌صورت تصادفی رشته‌ها را تولید می‌کند، به‌دست‌آمده‌اند.

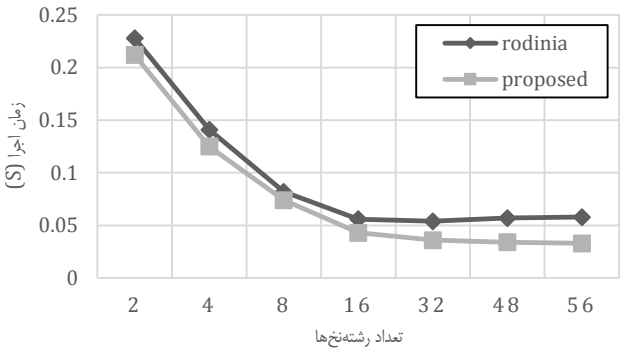
### ۵-۱- نتایج با استفاده از استراتژی بلوک‌بندی

در نمودار شکل ۹، مقایسه بین دو روش در هنگامی است که ابعاد آرایه ورودی مسئله برابر ۱۶۳۸۴ است. همچنین برای ابعاد ۸۱۹۲ و ۴۰۹۶ نیز مقایسه صورت گرفته و نمودار آن‌ها در شکل ۱۰ و شکل ۱۱ آمده است. در هر سه نمودار به ازای حالت‌های مختلف برای تعداد رشته‌نخ‌های به کار گرفته‌شده، سرعت اجرای روش ارائه شده بهتر است.

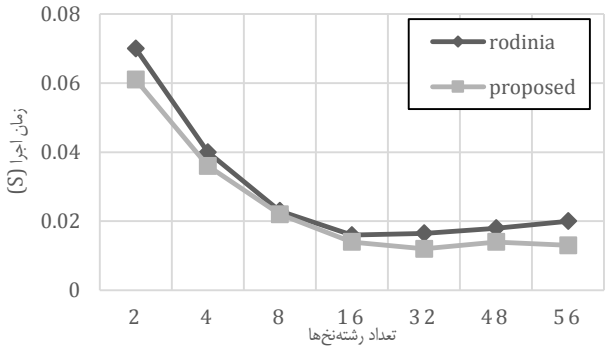
با مقایسه زمان اجرای دو روش در حالت‌ها مختلف، مشاهده می‌شود که روش ارائه شده تا حدود ۱,۹۸ برابر سریع‌تر از روش Rodinia عمل می‌کند.




شکل ۹ – مقایسه نتایج آزمایش دو روش بر روی رشته به طول ۱۶۳۸۴




شکل ۱۰– مقایسه زمان اجرای دو روش بر روی رشته به طول ۸۱۹۲



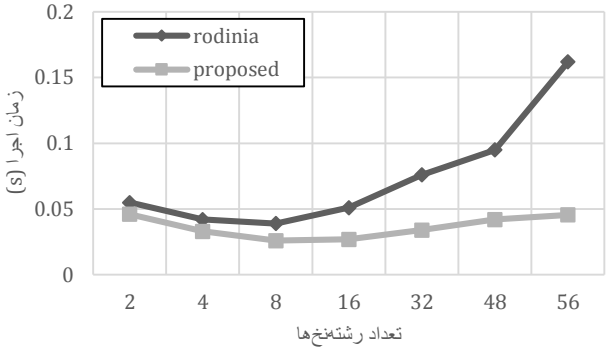

شکل ۱۱– مقایسه زمان اجرای دو روش بر روی رشته به طول ۴۰۹۶

### ۵-۲- نتایج بدون استفاده از استراتژی بلوک‌بندی

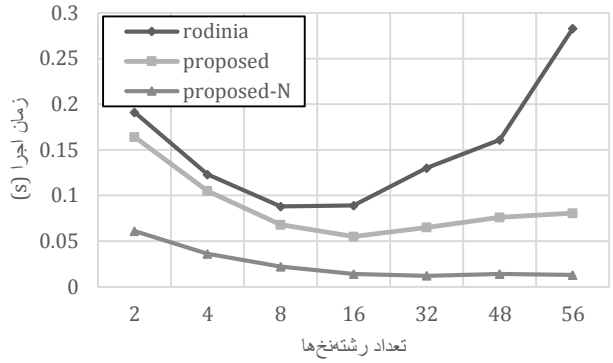
در پیاده‌سازی نسخه‌ی ۳,۱ از محک آزمون Rodinia ایده‌ی استراتژی بلوک‌بندی به مدل OPENMP آن اضافه شده است. از طرفی در روش پیشنهادی علاوه بر موازی‌سازی دوطرفه از ایده‌ی استراتژی بلوک‌بندی استفاده شده است. بدین ترتیب در این بخش به مقایسه‌ی چند حالت از پیاده‌سازی‌ها می‌پردازیم:

- مقایسه‌ی پیاده‌سازی روش ارائه شده (استفاده هم‌زمان از موازی‌سازی دوطرفه و استراتژی بلوک‌بندی) با پیاده‌سازی موجود در نسخه‌ی ۳,۱ از محک آزمون Rodinia .
- مقایسه روش‌ها در حالت عدم استفاده از بلوک‌بندی: یعنی روش ارائه شده در حالتی که از استراتژی بلوک‌بندی در آن استفاده نشده باشد (تنها موازی‌سازی دوطرفه در آن به‌کاررفته باشد) با نسخه‌ی ۳ از محک آزمون Rodinia مقایسه می‌شود.

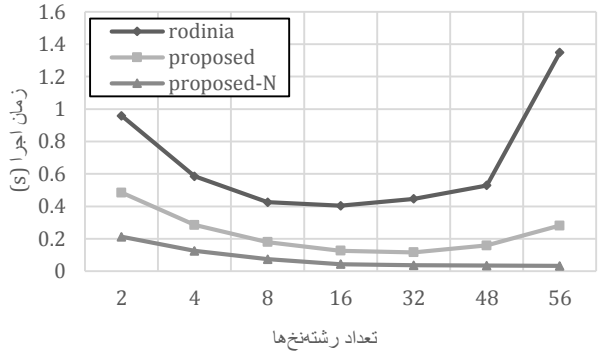
در شکل ۱۲ نتایج این آزمایش برای ابعاد ۲۰۴۸ مشاهده می‌شود. همان‌طور که مشاهده می‌شود، علاوه بر بهبود سرعت چشم‌گیر در روش ارائه شده نسبت به پیاده‌سازی Rodinia، در پیاده‌سازی Rodinia با افزایش تعداد رشته نخ‌ها از ۸ رشته نخ به ۱۶ رشته نخ، شاهد کاهش سرعت قابل‌ملاحظه‌ای هستیم و این نشان از استفاده‌ی نه‌چندان بهره‌مند از منابع در این روش دارد. این کاهش سرعت در اجرای الگوریتم برای ابعاد ۴۰۹۶ نیز وجود دارد که در شکل ۱۳ مشاهده می¬کنید. در نمودارهای مربوط به ابعاد ۴۰۹۶، ۸۱۹۲ و ۱۶۳۸۴ از مسئله (به ترتیب شکل ۱۳، شکل ۱۴ و بلوک‌بندی (که در شکل‌ها با برچسب Proposed-N نمایش داده شده است) نیز آورده شده است تا با مقایسه روش پیشنهادی در حالت‌های استفاده از استراتژی بلوک‌بندی وعدم استفاده از استراتژی بلوک‌بندی تأثیر به‌کارگیری بلوک‌بندی در روش پیشنهادی نیز مشخص گردد. در نتایج حاصل از اجرای پیاده‌سازی Rodinia بر روی ابعاد ۸۱۹۲ و ۱۶۳۸۴ از الگوریتم نیدلمن-وانچ، با افزایش تعداد هسته‌ها از ۱۶ هسته به ۳۲ هسته برخلاف روش ارائه شده، کاهش سرعت رخ می‌دهد که همچنان حاکی از این است که در روش ارائه شده به‌صورت بهره‌مندتری از منابع استفاده شده است.



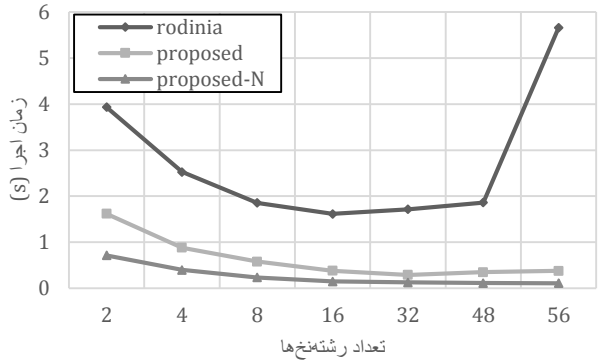

شکل ۱۲ – مقایسه زمان اجرای دو روش بدون استفاده از استراتژی بلوک‌بندی بر روی رشته به طول ۲۰۴۸




شکل ۱۳– نتایج زمان اجرای دو روش بدون بلوک‌بندی و روش ارائه شده با بر روی رشته به طول ۴۰۹۶ استراتژی بلوک‌بندی




شکل ۱۴ – مقایسه زمان اجرای دو روش بدون استفاده از استراتژی بلوک‌بندی و روش ارائه شده با استفاده از استراتژی بلوک‌بندی بر روی رشته به طول ۸۱۹۲




شکل ۱۵- نتایج زمان اجرای دو روش بدون بلوک‌بندی و روش ارائه شده با بر روی رشته به طول ۱۶۳۸۴ استراتژی بلوک‌بندی

با مقایسه‌ی زمان اجرای دو روش به ازای مقادیر مختلف تعداد هسته‌ها و ابعاد مسئله و در حالتی که در هیچ‌یک از روش‌ها تکنیک بلوک‌بندی استفاده نشود، مشاهده می‌شود که روش ارائه شده تا حدود ۵,۹ برابر سریع‌تر از روش Rodinia عمل می‌کند.

## ۶- نتیجه‌گیری

همان‌طور که ذکر شد از الگوریتم نیدلمن-وانچ برای حل یکی از مسائل پرکاربرد در زمینه بیوانفورماتیک یعنی انطباق رشته‌ها استفاده می‌شود و با توجه به محدودیت‌های ناشی از وابستگی‌های بین داده‌ای موجود در این الگوریتم (به هنگام استفاده از جداول امتیاز نامتقارنی نظیر BLOSUM62)، موازی‌سازی به‌صورت قطری در هر گام می‌تواند مفید واقع گردد. اما، تنها استفاده از این روش حداکثر میزان بهره‌وری از پردازنده‌های چندهسته‌ای را ایجاد نمی‌کند. ازاین‌رو در این کار با تقسیم فضای مسئله و ارائه رویکرد موازی‌سازی دوطرفه‌ی عملیات بر پایه‌ی نگاشت آرایه‌ی به‌کاررفته در الگوریتم نیدلمن-وانچ به یک گراف، برای به دست آوردن امتیاز بیشینه حاصل از انطباق، می‌توان بهبود عملکردی تا حدود ۵,۹ برابر و نسبت به بهترین پیاده‌سازی موجود با استفاده از روش بلوک‌بندی بهبودی در حد ۱,۹۸ برابر کسب کرد. ازآنجاکه می‌توان از عملیات انطباق یک جفت رشته در انطباق‌های چندگانه نیز استفاده نمود می‌توان با به‌کارگیری رویکرد ارائه شده در انطباق‌های چندگانه نیز بهبود سرعت را شاهد بود. همچنین با استفاده از این روش، می‌توان از بیشترین میزان کارایی پردازنده‌ها در راستای استفاده بهینه از انرژی مصرفی با تعداد هسته‌های بالا بهره برد.

## تشکر و قدردانی

از مرکز پردازش سریعِ پژوهشگاه دانش‌های بنیادی که امکانات موردنیاز را برای انجام این پروژه فراهم آورد، کمال تشکر و قدردانی را داریم.

## ۷- مراجع

[1] J. Tong, J. Pei and N. V. Grishin, "SFESA: a web server for pairwise alignment refinement by secondary structure shifts," *BMC bioinformatics,* vol. 16, p. 282, 2015.

[2] "A One-Letter Notation for Amino Acid Sequences," *IUPAC-IUB Commission on Biochemical Nomenclature (CBN),* vol. 5, no. 2, p. 151–153, 1968.

[3] F.Sanger, "The Arrangement of Amino Acids in Proteins," *Advances in Protein Chemistry and Structural Biology,* vol. 7, pp. 1-67, 1952.

[4] R. Aasland, C. Abrams, C. Ampe, L. J. Ball, M. T. Bedford, G. Cesareni, M. Gimona, J. H. Hurley, T. Jarchau, V.-P. Lehto, M. A. Lemmon, R. Linding, B. J. Mayer, M. Nagai, M. Sudol, U. Walter and Steve, "Normalization of nomenclature for peptide motifs as ligands of modular protein domains," *FEBS Letters,* vol. 513, no. 1, pp. 141-144, 2002.

[5] M. J. Bower, F. Cohen and R. Dunbrack, "Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a new homology modeling tool1," *Journal of Molecular Biology,* vol. 267, pp. 1268-1282, 1997.

[6] S. Gálvez, D. Díaz, P. Hernández, F. J. Esteban, J. A. Caballero and G. Dorado, "Next-Generation Bioinformatics: Using Many-Core Processor Architecture to Develop a Web Service for Sequence Alignment," *Bioinformatics,* vol. 26, no. 5, p. 683–686, 2010.

[7] S. Che, J. W. Sheaffer, M. Boyer, L. G. Szafaryn, L. Wang and K. Skadron, "A Characterization of the Rodinia Benchmark Suite with Comparison to Contemporary CMP Workloads," In *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC'10),* pp. 1-11, 2010.

[8] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee and K. Skadron, "Rodinia: A Benchmark Suite for Heterogeneous Computing," *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on,* pp. 44-54, 2009.

[9] R. Chenna, H. Sugawara, T. Koike, R. Lopez, T. J. Gibson, D. G. Higgins and J. D. Thompson, "Multiple sequence alignment with the Clustal series of programs," *Nucleic Acids Research,* vol. 31, no. 13, pp. 3497-3500, 2003.

[10] V. O. Polyanovsky, M. A. Roytberg and V. G. Tumanyan, "Comparative analysis of the quality of a global algorithm and a local algorithm for alignment of two sequences," *Algorithms for Molecular Biology,* vol. 6, p. 25, 2011.

[11] S. A.Shehab, A. Keshk and H. Mahgoub, "Fast Dynamic Algorithm for Sequence Alignment based on Bioinformatics," *International Journal of Computer Applications,* vol. 37, pp. 54-61, 2012.

[12] X. Xia, *Bioinformatics and the Cell.* Springer, Boston, MA, pp. 23-48, 2007.

[13] S. B. NEEDLEMAN and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology,* vol. 48, no. 3, pp. 443–453, 1970.

[14] S. HENIKOFF and J. G. HENIKOFF, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 89, no. 22, pp. 10915-10919, 1992.

[15] D. S. Hirschberg, "A linear space algorithm for computing maximal common subsequences," *Communications of the ACM,* vol. 18, no. 6, pp. 341-343, 1975.

[16] S. Maleki, M. Musuvathi and T. Mytkowicz, "Parallelizing dynamic programming through rank convergence," *PPoPP '14 Proceedings of the 19th ACM SIGPLAN symposium on Principles and practice of parallel programming,* vol. 49, no. 8, pp. 219-232, 2014.

[17] S. Maleki, M. Musuvathi and T. Mytkowicz, "Efficient parallelization using rank convergence in dynamic programming algorithms," *Communications of the ACM,* vol. 59, pp. 85-92, 2016.

[18] S. A. Manavski and G. Valle, "CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment," *BMC bioinformatics,* vol. 9, 2008.

[19] H. Hu and Z. Ji, "Parallelization of Needleman-Wunsch Algorithm Based on Software Pipelining," *International Journal of Engineering and Manufacturing (IJEM),* vol. 1, pp. 59-64, 2011.

[20] D. Li and M. Becchi, "Multiple Pairwise Sequence Alignments with the Needleman-Wunsch Algorithm on GPU," in *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:*, Salt Lake City, UT, 2012.

[21] T. R. P. Siriwardena and D. N. Ranasinghe, "Accelerating global sequence alignment using CUDA compatible multi-core GPU," *Information and Automation for Sustainability (ICIAFs), 2010 5th International Conference on,* pp. 201-206, 2010.

[22] O. Trelles-Salazar, E. Zapata and J. Carazo, "On an efficient parallelization of exhaustive sequence comparison algorithms on message passing architectures," *Bioinformatics,* vol. 10, no. 5, pp. 509-511, 1994.

[23] S. Vinogradov, J. Fedorova, D. Curran, S. McIntosh-Smith and J. Cownie, "OpenMP 4.0 vs. OpenCL: Performance comparison," in *Paper presented at The OpenMP Developers Conference (OpenMPCon)*, Aachen, Germany, 2015.

[1] Homology modeling
[2] Active site prediction
[3] Next-Generation sequencing
[4] Local

[5] Global
[6] GPU
[7] Paramid parallel computer

**میلاد غفوری** تحصیلات دانشگاهی خود را در مقطع کارشناسی در مهندسی کامپیوتر-نرم‌افزار در سال ۱۳۹۵ در دانشگاه خوارزمی تهران و در مقطع کارشناسی ارشد در رشته مهندسی کامپیوتر-هوش مصنوعی در سال ۱۳۹۸ در دانشگاه علم و صنعت ایران به پایان رسانده است. زمینه‌های تحقیقاتی مورد علاقه ایشان طراحی الگوریتم، هوش مصنوعی، علم داده، پردازش زبان طبیعی و بیوانفورماتیک است.

آدرس پست الکترونیکی ایشان عبارت است:

miladghfouri0@gmail.com

**آرمین احمدزاده** در حال حاضر دانشجوی دکتری مهندسی کامپیوتر (گرایش معماری کامپیوتر) در دانشگاه صنعتی شریف است. وی مدرک کارشناسی ارشد خود را در همین رشته-گرایش در سال ۱۳۹۳ از دانشگاه آزاد دریافت نموده است. زمینه‌های تحقیقاتی مورد علاقه ایشان معماری کامپیوتر، پردازش موازی، سیستم‌های توزیع شده و پردازنده های گرافیکی است.

آدرس پست الکترونیکی ایشان عبارت است:

a.ahmadzadeh@ipm.ir

**سعید گرگین** مدرک کارشناسی و کارشناسی ارشد مهندسی کامپیوتر خود را به ترتیب از دانشگاه آزاد اسلامی تهران جنوب و علوم و تحقیقات تهران و مدرک دکتری معماری کامپیوتر را از دانشگاه شهید بهشتی، در سال ۱۳۸۹ دریافت کرد. وی هم اکنون استادیار مهندسی کامپیوتر در گروه فناوری اطلاعات و سیستم های هوشمند سازمان پژوهش های علمی و صنعتی ایران است. علایق تحقیقاتی ایشان شامل حساب کامپیوتری و الگوریتم‌های محاسباتی، سیستم های با قابلیت پیکربندی مجدد، پردازش سریع و طراحی VLSI می‌باشد

آدرس پست الکترونیکی ایشان عبارت است:

gorgin@irost.ir

The CSI Journal on Computing Science and Information Technology
Vol. 17, No.1, 2019

# Efficient parallelization and high-performance of sequences alignment on the multi-core platforms

**Milad Ghafouri[1], Armin Ahmadzadeh[2], Saeid Gorgin[3]**

[1, 2, 3] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[2] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
[3] Iranian Research Organization for Science and Technology (IROST), Tehran, Iran

## Abstract

Sequence alignment is a widely used process in bioinformatics applications that is used to find the similarities among amino-acid or DNA sequences. Pairwise sequences alignment is a fundamental operation that is also worked in multiple sequences alignments. The Needleman–Wunsch algorithm is one of the sequence alignment approaches which uses a dynamic programming method. One of the challenges in this algorithm is high time complexity. To mitigate this issue and improve the execution time, parallel solutions can be considered. Moreover, with recent multi-core processors and the possibility of performing efficient computations in parallel, significant acceleration can be achieved. In the existing parallelization methods, in each step of dynamic programing, the array cells of one diameter, in a two-dimensional array, are completed simultaneously and filled in a parallel manner. However, these methods cannot efficiently utilize the processor's resources. Therefore, in this paper, by changing the perspective on the problem and considering a graph-like perception, a method is suggested to efficiently utilize the processor cores and provide a suitable performance compared to previous methods. The results show that in the proposed implementation, performance improvement is achieved up to 5.9X compared to the previous works.

**Keywords**: Needleman-Wunsch algorithm, Sequence alignment, bioinformatics, Multi-Core, Parallel